Princeton University
COS 217: Introduction to Programming Systems
Fall 2002 Final Exam Answers

Question 1

```
   .section ".text"

!-----------------------------------
! int f(int *a)
!
! Register map
!   %i0  int *a
!   %l0  int i;
!   %l1  int s;
!-----------------------------------

   .align 4
   .global f
f:

   save %sp, -96, %sp

   ! s = 0;
   clr  %l1

   ! i = 0;
   clr  %l0

loop:

   ! if (a[i] == 0) goto endloop;
   sll  %l0, 2, %l2
   ld   [%i0 + %l2], %l3
   cmp  %l3, 0
   be   endloop
   nop

   ! s += a[i];
   add  %l1, %l3, %l1

   ! i++;
   inc  %l0

   ! goto loop;
   ba   loop
   nop

endloop:

   ! return s;
   mov  %l1, %i0
   ret
   restore
```

Question 2

```
enum ExprType {EXPR_FUNC, EXPR_ADD, EXPR_VALUE};

typedef struct Expr *Expr_T;

struct Expr
{
   enum ExprType iType;
   union
   {
      struct {char *pcFunc; Expr_T oExpr1; } sFuncExpr;
      struct {Expr_T oExpr1; Expr_T oExpr2; } sAddExpr;
      struct {double dValue; } sValue;
   } u;
};


Expr_T Expr_newFuncExpr(char *pcFunc, Expr_T oExpr1)
{
   Expr_T oExpr;
   assert(pcFunc != NULL);
   assert(oExpr1 != NULL);
   oExpr = (Expr_T)malloc(sizeof(*oExpr));
   assert(oExpr != NULL);
   oExpr->iType = EXPR_FUNC;
   oExpr->u.sFuncExpr.pcFunc = pcFunc;
   oExpr->u.sFuncExpr.oExpr1 = oExpr1;
   return oExpr;
}

Expr_T Expr_newAddExpr(Expr_T oExpr1, Expr_T oExpr2)
{
   Expr_T oExpr;
   assert(oExpr1 != NULL);
   assert(oExpr2 != NULL);
   oExpr = (Expr_T)malloc(sizeof(*oExpr));
   assert(oExpr != NULL);
   oExpr->iType = EXPR_ADD;
   oExpr->u.sAddExpr.oExpr1 = oExpr1;
   oExpr->u.sAddExpr.oExpr2 = oExpr2;
   return oExpr;
}

Expr_T Expr_newValueExpr(double dValue)
{
   Expr_T oExpr;
   oExpr = (Expr_T)malloc(sizeof(*oExpr));
   assert(oExpr != NULL);
   oExpr->iType = EXPR_VALUE;
   oExpr->u.sValue.dValue = dValue;
   return oExpr;
}
```

Question 3

```
struct Instr
{
   unsigned int uiOp: 2;
   unsigned int uiRd: 5;
   unsigned int uiOp3: 6;
   unsigned int uiRest: 19;
};

int anydest(unsigned int code[], int N, int d)
{
   int i;
   struct Instr sInstr;

   for (i = 0; i < N; i++)
   {
      sInstr = *(struct Instr *)(code + i);
      if ((sInstr.uiOp == 2) && (sInstr.uiRd == d) && (sInstr.uiOp3 == 16))
         return 1;
   }
   return 0;
}
```

Alternate answer:

```
int anydest(unsigned int code[], int N, int d)
{
   int i;
   unsigned int uiOp, uiRd, uiOp3;
   for (i = 0; i < N; i++)
   {
      uiOp = code[i] >> 30;
      uiRd = (code[i] << 2) >> 27;
      uiOp3 = (code[i] << 7) >> 26;
      if ((uiOp == 2) && (uiRd == d) && (uiOp3 == 16))
         return 1;
   }
   return 0;
}
```

Question 4

matrix.h

```
#ifndef MATRIX_INCLUDED
#define MATRIX_INCLUDED

#define N 1000

typedef struct Matrix *Matrix_T;

Matrix_T Matrix_new(void);
void Matrix_free(Matrix_T oMatrix);
void Matrix_incr(Matrix_T oMatrix, int iRow, int iCol);
int Matrix_getRowSum(Matrix_T oMatrix, int iRow);
int Matrix_getColSum(Matrix_T oMatrix, int iCol);

#endif
```

matrix.c

```
#include "matrix.h"
#include <stdlib.h>
#include <assert.h>

struct Matrix
{
   int matrix[N][N];
};

Matrix_T Matrix_new(void)
{
   Matrix_T oMatrix;
   oMatrix = (Matrix_T)calloc(1, sizeof(*oMatrix));
   return oMatrix;
}

void Matrix_free(Matrix_T oMatrix)
{
   free(oMatrix);
}

void Matrix_incr(Matrix_T oMatrix, int iRow, int iCol)
{
   assert(oMatrix != NULL);
   assert(iRow >= 0);
   assert(iRow < N);
   assert(iCol >= 0);
   assert(iCol < N);
   oMatrix->matrix[iRow][iCol]++;
}

int Matrix_getRowSum(Matrix_T oMatrix, int iRow)
{
   int iCol;
   int iSum = 0;
   assert(oMatrix != NULL);
   assert(iRow >= 0);
   assert(iRow < N);
   for (iCol = 0; iCol < N; iCol++)
     iSum += oMatrix->matrix[iRow][iCol];
   return iSum;
}

int Matrix_getColSum(Matrix_T oMatrix, int iCol)
{
   int iRow;
   int iSum = 0;
   assert(oMatrix != NULL);
```

```
   assert(iCol >= 0);
   assert(iCol < N);
   for (iRow = 0; iRow < N; iRow++)
     iSum += oMatrix->matrix[iRow][iCol];
   return iSum;
}
```

client.c

```
#include "matrix.h"
#include <stdio.h>

int main(int argc, char ** argv)
{
   int i, j;
   int rowSum, colSum;
   int total = 0;
   Matrix_T oMatrix;

   oMatrix = Matrix_new();
   while (2 == scanf("%d %d", &i, &j))
      Matrix_incr(oMatrix, i, j);

   for (i = 0; i < N; i++)
   {
      colSum = Matrix_getColSum(oMatrix, i);
      total += colSum * colSum;
   }

   for (j = 0; j < N; j++)
   {
      rowSum = Matrix_getRowSum(oMatrix, j);
      total += rowSum * rowSum;
   }

   printf("%d\n", total);
   Matrix_free(oMatrix);
   return 0;
}
```

fastmatrix.c

```
#include "matrix.h"
#include <stdlib.h>
#include <assert.h>

struct Matrix
{
   int rowSum[N];
   int colSum[N];
};

Matrix_T Matrix_new(void)
{
   Matrix_T oMatrix;
   oMatrix = (Matrix_T)calloc(1, sizeof(*oMatrix));
   return oMatrix;
}

void Matrix_free(Matrix_T oMatrix)
{
   free(oMatrix);
}

void Matrix_incr(Matrix_T oMatrix, int iRow, int iCol)
{
   assert(oMatrix != NULL);
   assert(iRow >= 0);
   assert(iRow < N);
```

```
   assert(iCol >= 0);
   assert(iCol < N);
   oMatrix->rowSum[iRow]++;
   oMatrix->colSum[iCol]++;
}

int Matrix_getRowSum(Matrix_T oMatrix, int iRow)
{
   assert(oMatrix != NULL);
   assert(iRow >= 0);
   assert(iRow < N);
   return oMatrix->rowSum[iRow];
}

int Matrix_getColSum(Matrix_T oMatrix, int iCol)
{
   assert(oMatrix != NULL);
   assert(iCol >= 0);
   assert(iCol < N);
   return oMatrix->colSum[iCol];
}
```

Question 5

Part A

s2	s1	s0	a	b	c	d	e	f	g
0	0	0	1	1	1	1	1	1	0
0	0	1	0	1	1	0	0	0	0
0	1	0	1	1	0	1	1	0	1
0	1	1	1	1	1	1	0	0	1
1	0	0	0	1	1	0	0	1	1
1	0	1	1	0	1	1	0	1	1
1	1	0	1	0	1	1	1	1	1
1	1	1	1	1	1	0	0	0	0

Part B

(Cannot draw a circuit diagram using text. Here are the Boolean algebra formulas.)

```
a = (~s2 & ~s1 & ~s0) | (~s2 & s1 & ~s0) | (~s2 & s1 & s0) | (s2 & ~s1 & s0) |
    (s2 & s1 & ~s0) | (s2 & s1 & s0)

b = (~s2 & ~s1 & ~s0) | (~s2 & ~s1 & s0) | (~s2 & s1 & ~s0) | (~s2 & s1 & s0) |
    (s2 & ~s1 & ~s0) | (s2 & s1 & s0)

c = (~s2 & ~s1 & ~s0) | (~s2 & ~s1 & s0) | (~s2 & s1 & s0) | (s2 & ~s1 & ~s0) |
    (s2 & ~s1 & s0) | (s2 & s1 & ~s0) | (s2 & s1 & s0)
```

Part C

The 7-segment displays show the number “217.”

Question 6

```
int sameinorder(Tree t1, Tree t2)
{
   int pid1, pid2, result;
   int pipe1[2];
   int pipe2[2];

   pipe(pipe1);
   pipe(pipe2);

   pid1 = fork();
   if (pid1 == 0)
   {
      close(pipe1[0]);
      close(pipe2[0]);
      close(pipe2[1]);
      putleaves(pipe1[1], t1);
      exit(0);
   }

   pid2 = fork();
   if (pid2 == 0)
   {
      close(pipe1[0]);
      close(pipe1[1]);
      close(pipe2[0]);
      putleaves(pipe2[1], t2);
      exit(0);
   }

   close(pipe1[1]);
   close(pipe2[1]);

   result = compare(pipe1[0], pipe2[0]);

   close(pipe1[0]);
   close(pipe2[0]);

   kill(pid1, SIGKILL);
   kill(pid2, SIGKILL);

   wait(NULL);
   wait(NULL);

   return result;
}
```